In the Specification:

On page 15, please replace the paragraph beginning on line 4 with the following amended paragraph:

A file encryption key is stored in the file header **402** encrypted under a personal key *k* belonging to the user, derived from a user-supplied "passphrase/password." Such an encrypted file encryption key **406** is illustrated as $Enc_k(ke, ki, hash(ke, ki))$ in **Figure 4**. The file encryption key may also be encrypted with a public key or keys *pk* of one or more trusted third parties ~~an~~ and incorporated in the file header **402** so as to allow third parties access to the file contents. Such an encrypted file encryption key **410** is illustrated as $Enc_{pk}(ke, ki, hash(ke, ki))$ in **Figure 4**. A message authentication code (MAC) **408** may also be included within the file header **402** to verify the authenticity of the file after decryption. Therefore, a file can be decrypted by the user who created the encrypted file, and who has the "passphrase" needed to decrypt the file, and/or by a trusted third party that the creating-user has granted access to the file.

On page 17, please replace the paragraph beginning on line 22 with the following amended paragraph:

The operations for file management illustrated in **Figures 5** through **12** may provide core functions which provide for the control of encrypted files. Thus, as will be appreciated by those of skill in the art in light of the present disclosure, these core functions may be further manipulated to provide additional file management functions. For example, a rename file operation could be associated with a file by retrieving the file and storing the file with the new file name. Optionally, the old file could be overwritten, deleted, or marked as inaccessible. Similarly, third party access to files and file ownership may be changed through the retrieval, storage and password or key change operations. Thus, for example, to change which users are trusted third party's, the public

key operations ~~descried~~ described below could be utilized to replace one third ~~parties~~ party's public key with ~~anothers~~ another's and, thereby, provide access to file to a different third party. Similarly, file ownership could be changed by allowing the new owner third party access to the file, the new owner could retrieve the file and store the file under a new tuple (*id, fid*) and, thereby, obtain ownership of the file. Thus, the core functions described herein may provide a robust feature set which may be readily expanded through combinations of operations. Furthermore, additional "file management" functions could also be incorporated, for example, utilizing conventional file server operations.

On page 26, please replace the paragraph beginning on line 12 with the following amended paragraph:

When the personal key client receives the response from the file server, if the response is negative, operations of **Figure 8 9** may terminate at block **704** and, optionally, the error may be reported to the user. However, if the response provides the encrypted file and file header, the personal key client extracts the encrypted value associated with the trusted third party from the header (block **708**), for example, $Enc_{pk}(ke, ki, Hash(ke, ki))$. The extracted encrypted value is decrypted with the trusted third party's private key *sk* to recover the encryption key utilized to encrypt the file (block 710). For example, in the embodiment illustrated in **Figure 9**, the values *ke*, *ki* and Hash(*ke, ki*) may be recovered. That is, $ke, ki, Hash(ke, ki) = Dec_{sk}(Enc_{pk}(ke, ki, Hash(ke, ki)))$.